Sensores:

* ***Intel RealSense R200 RGB-D Camera***
* ***Mini PC NUC*** → processa dados para gerar mapa 3D
* ***Flight Controler*** → GPS e IMU (Responsável por: Gyroscope, Accelerometer, Barometer, e Magnetometer para efetuar a navigation.

Packages ROS:

* GMapping
* ***Pointcloud\_to\_Laserscan:*** create a virtual laserscan from your RGBD device. Daqui podemos tirar o ângulo de rotação máximo, o alcance máximo e a diferença entre o ângulo máximo e minimo.

(<http://wiki.ros.org/pointcloud_to_laserscan>)

Comandos Úteis:

* ***/head\_camera\_link topic:*** Obter a posição certa do robot em x,y,z (dado o nosso problema não ser a localização mas sim o mapping).
* ***euler\_from\_quaternion:*** Obter orientação do robot em graus.
* ***node mbot\_2dnav:*** Converte um mapa 3D para 2D.

Algoritmos:

* ***Occupancy Grid Mapping:*** Divide as células com a mesma dimensão e cada célula tem uma correspondente probabilidade de ocupação. cria-se uma circunferência de raio alpha e centro o robot para facilitar medições de distâncias
  + ***The log odds function:*** evitar que as probabilidades perto de um ou zero se tornem instáveis.
  + ***Inverse Range-Sensor Model:*** determinar a ocupação de cada célula com base na distância do robot a cada célula:

1. célula fora do range do robot célula fica com valor 0 -> unknown
2. célula dentro do alcance do robot e perto de fronteira de obstáculo célula fica com valor 1-> temos obstáculo
3. caso contrário temos célula livre